Linus Ververs
AG Software Engineering
Institut für Informatik
Fachbereich Mathematik & Informatik

Freie Universität Berlin

Proseminar: Software Engineering in Society

Qualität wiss. Arbeiten & Literaturrecherche

Plan für heutige Stunde

1. Wiederholung und Fazit zu „A Tale from the Trenches: Cognitive Biases and Software Development”
   ca. 20-30 min

2. Gruppenarbeit: Literaturrecherche
   ca. 60-70 min

Hinweis zu den Vortragsfolien: Bitte am Tag vor der eigenen Präsentation hochladen in den dedizierten Ordner im kvv.

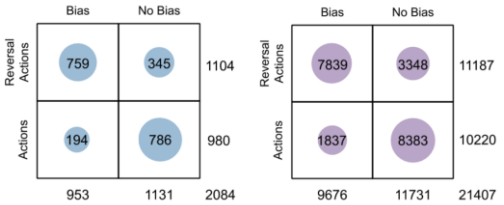
Zusammenfassung: Ergebnissektion

1. Presence of Biases in Developer Actions ✓
   - Klassifizierung in “Reversal Actions”/ “Actions” & “Bias” / “No Bias”
2. Distribution of Biases ✓
   - Zuordnung der Aktionen in 10 Bias-Kategorien
3. Consequences of Biases
   - Einteilung der Effekte von Bias-Aktionen in 4 Kategorien mittels negotiated agreement
4. Dealing with Consequences of Biases
   - Interviews mit 16 Entwicklern zur Häufigkeit von Biases und Tipps zur Vorbeugung

Ergebnisse aus: Chattopadhyay, Souti, et al. "A tale from the trenches: cognitive biases and software development." *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. 2020.

Zentrale Ergebnisse (ohne Interviews)

1. “Not only are biases frequently present during development (45.72%), but also biased actions are significantly more likely to be reversed later. Furthermore, developers spend a significant amount of time reversing these biased actions.”
2. “Bias categories affected developers uniquely, causing reversal actions. Fixation(CB3) was the most common and were associated with the most number of reversals, followed by convenience(CB6) and subconscious action(CB7) biases. Memory(CB10) and superficial selection(CB9) biases had the least number of reversals.”
3. “Biases affect multiple aspects of problem solving during development. Specifically, biases affect how adequately developers explore the solution space, how thoroughly they engage in sensemaking, how effectively they retain context, and how efficiently they invest their attention.”



(a) Distribution of Actions (b) Distribution of Time (sec)

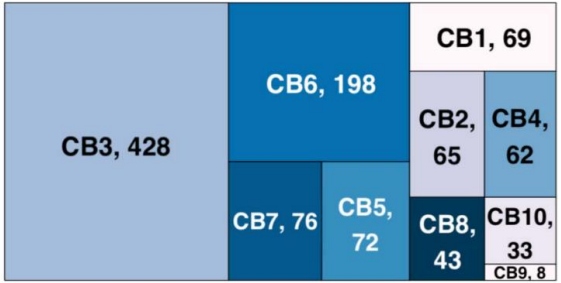


Table 5: Consequences of Biases

Consequence	Bias Categories (CB)
Inadequate Exploration	1, 2, 4, 5, 6, 10
Reduced Sense-making	5, 6, 7, 8, 9
Preserving Context	3, 5, 8
Misplaced Attention	3, 4, 8, 9

Zitate und Grafiken aus: Chattopadhyay, Souti, et al. "A tale from the trenches: cognitive biases and software development." *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. 2020.

Ergebnisse Interviews

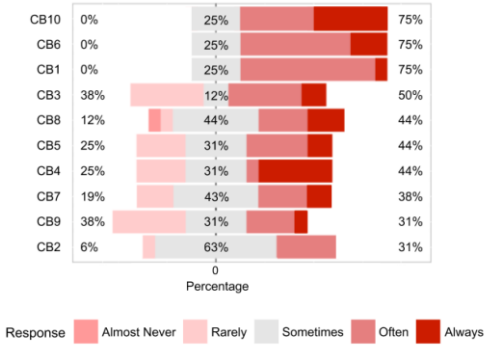


Figure 3: Perceived Frequency of Biases from Interviews ranging from 'Almost Never' to 'Always'. Bias categories are ordered in descending order from most to least frequent (shown as percentages).

Im Vergleich zu Ergebnissen der Beobachtung:

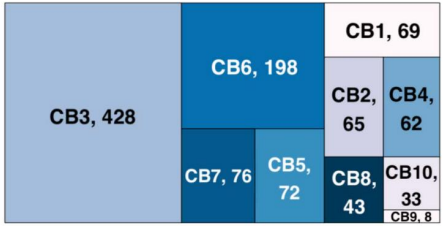


Table 6: Helpful Practices

Categories	Subcategories	Time	Who	Biases	Tools
Stepping Back	**Incentivized Training:** discussion of clean code benefits, long term goals	B/D	T/I	2,6,7	NA
	Non-Code Days: documentation days, test fest, familiarize with concepts	D	T	1,6,10	
	Meaningful Configurations, updating configurations, meaningful defaults	B	T	4,10	
Different Perspectives	**Confer With Developers:** pair programming, collaborative decision brainstorming (code/design/tool), verify global changes, designated tool guy	B/D	T	1,2,3,6,9,10	Slack, Hipchat
	Open Communication: encourage communication, communicate early with teams like QA. Promote focus on functionality and need	D	T	2,5,6,7	
Systematic Approach	**Systematic Exploration:** prior research on tools, compare & contrast solutions, problem decomposition	B	I	1,2,3,4,5, 6,7,8,9	Dev Tools, Sonarlint
	Big Picture in Mind: re usability of code, backwards compatibility	D	I	1,2,6,10	
	Consistent Early Feedback: reviews (design, expert, peer), sprint meetings	D/A	T	1,2,3,4,5,6	
RTFM	**Reference Doc.:** req/API/design doc, code comments, online sources	B	T	1,4,5	IDE Suggestion
	Journal Options/Alternatives: playbook, team diary	B	T/I	1,2,10	
	Meaningful & Relevant Specifications: standard specs, severity/relevant levels for warnings, protocol for resolution of warnings, descriptive errors	B/D	T	2,4,7,8	
Processes	**SE Concepts:** agile, code review, shared artifacts, reduced ownership, design first, UML diagrams, user story, TDD, BDD, constant debugging, data flows	B/D/A	T/I	1,3,4,5,6, 7,8,9,10	ZenHub, Gerrit, Debugger, IDE, JIRA, JaCoCo
	Standardization: corporate/coding/package standards, right arch. & microservices, clean code, performance test, impact analysis	A	T	1,2,5,6,7,8, 9,10	
	Problem Solving Strategies: divergence & convergence thinking, defensive programming, negative hypothesis testing, timebox, note todos in code	B/D/A	I	1,3,5,6,7,8,9	

Zitate und Grafiken aus: Chattopadhyay, Souti, et al. "A tale from the trenches: cognitive biases and software development." *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. 2020.

Fazit zum Artikel

Gemeinsames Lesen der „Threats to validity“

Persönliche Einschätzung:

- Spannender und guter Artikel, aber wie die Autoren sagen:
  „This study reveals the tip-of-the-iceberg […]”

- Hauptkritikpunkte:
  - Fokussierung auf „Actions“ statt der kritischen Entscheidungspunkte (Relevanz)
  - Warum keine Transparenz beim Agreement bei der Kategorisierung der Biases? (Glaubwürdigkeit)
  - Think-aloud-Methode schränkt „Praxisbezug“ stark ein (Relevanz)

Zitate und Grafiken aus: Chattopadhyay, Souti, et al. "A tale from the trenches: cognitive biases and software development." *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. 2020.

Zweiter Teil: Literaturrecherche

Angenommen wir interessieren uns allgemein für **Entscheidungsfindung im Software Engineering** und sind als erstes auf diesen Artikel („A tale from the trenches: cognitive biases and software development”) gestoßen: **Wie finden wir davon ausgehend weitere relevante Arbeiten?**

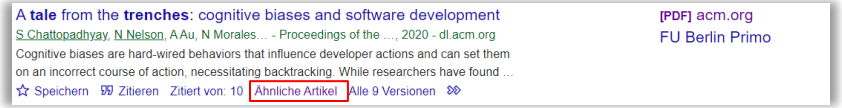
1. Stichwortsuche über Google Scholar
2. Von ursprünglichem Artikel zitierte Arbeiten
3. „Ähnliche Artikel“ über Google Scholar
4. Andere Artikel der gleichen Konferenz

5. (Artikel, die diesen Artikel zitiert haben) (← Machen wir heute nicht)

Gruppenarbeit: Literaturrecherche

Jede Gruppe sucht 10 zur Fragestellung passende Artikel mithilfe der Methode. Bewertung der Relevanz des Artikels nach Lesen des Abstracts von 1-5 Sternen. Anschließend neue Gruppen und Vergleich der gefundenen Arbeiten.

1. Stichwortsuche über Google Scholar
2. Von ursprünglichem Artikel zitierte Arbeiten
3. „Ähnliche Artikel“ über Google Scholar
4. Artikel, die auf gleicher Konferenz erschienen sind: [International Conference on Software Engineering](#) (abrufbar computer.org)



Fragestellung: Angenommen wir interessieren uns allgemein für **Entscheidungsfindung im Software Engineering** und sind als erstes auf diesen Artikel („A tale from the trenches: cognitive biases and software development”) gestoßen: **Wie finden wir davon ausgehend weitere relevante Arbeiten?**

Lernziele - Proseminar

- Wie baue ich eine wissenschaftliche Arbeit auf?
- Wie schätze ich die Qualität wissenschaftlicher Arbeiten ein? ✓
- Wie findet man gute Quellen und zitiert diese richtig? ✓
- Wir schreibt man eine wissenschaftliche Arbeit in LaTeX?
- Wie bereitet man ein wissenschaftliches Thema in einem Vortrag auf?

→ Am Ende: Gerüstet für die Bachelorarbeit